Simulation Lab #4:
Dynamic Modeling and Simulation of Muscle-Tendon Actuators

Laboratory Developers: Darryl Thelen, Silvia Blemker, Clay Anderson, Scott Delp

ME 382: Modeling and Simulation of Human Movement
Professor Scott Delp
Stanford University

Spring 2001

I.   Introduction

The force producing properties of muscle are complex, highly nonlinear and can have substantial effects on movement (See McMahon 1984 for review). For simplicity, lumped-parameter dimensionless muscle models, capable of representing a range of muscles with different architectures, are commonly used in the dynamic simulation of movement (Zajac, 1989). In this tutorial, we will review the differential equations that describe muscle activation and muscle-tendon contraction dynamics when using a Hill-type muscle model. You will use SIMM/Pipeline routines to implement muscle-tendon models and conduct some simulations to investigate how various model parameters can affect the dynamic response of the actuators. The lab will conclude with a *Virtual Muscle Tug-of-War* in which you will design an *optimal* muscle and compete directly against others in the class– may the best muscle win.

II.   Objectives

The purpose of this tutorial is to learn how to use and modify SIMM Pipeline routines to model and simulate muscle-tendon dynamic contractions.

By working through this tutorial, you will:

- Become familiar with the differential equations describing muscle activation and muscle-tendon contraction dynamics

- Learn how to model and simulate dynamic musculo-tendon actions using SIMM Pipeline routines

- Become comfortable with modifying existing code that models muscle activation and mechanics

- Explore the effect of various model parameters and simulation conditions on the dynamic response of muscle

- Design your own 'optimal' musculo-tendon actuator to compete in a virtual muscle tug of war

III. Deliverables

Turn in computer files from your modeling and simulation work to `/home/me382/`*`username`*`/L4`, where *`username`* is your workstation login. You are the only person who will have read and write permission to this directory during the week or two that this lab is in progress. In addition, hand in a written report that summarizes your findings and addresses the questions that are posed.

Please turn in:

1. A written report

    A Microsoft Word template for the report called `lab4_report.doc` is available on the BME workstations in `/software/nmbl/tutorials/me382/L4`.

    Deposit the following computer files to `/home/me382/`*`username`*`/L4`

2.   `username.msl`   Muscle file to be used in virtual muscle tug of war, where `username` is your BME login ID

IV. Input Files Needed

1. SIMM Files

`/software/nmbl/tutorials/me382/L4/tug_of_war.jnt`
SIMM joint file used to animate the translating block simulations

`/software/nmbl/tutorials/me382/L4/tug_of_war.msl`
SIMM muscle file that contains parametric descriptions of the two muscles included in the translating block simulations

`/software/nmbl/tutorials/me382/L4/bones/small_box.asc`
bone file for a 10 cm x 10 cm box

`/software/nmbl/tutorials/me382/L4/bones/floor1s.asc`
`/software/nmbl/tutorials/me382/L4/bones/floor2s.asc`
bone files for the floor

2. Pipeline Code

Main Driver Routine for Forward Dynamic Simulations

`/software/simm2/pipeline/src/formain.c`
This file contains the main driver routine for a forward dynamic simulation, and several other utility routines that are independent of the musculoskeletal model in the simulation. This file can be generated by SIMM/Pipeline. You will probably want to generate it once and then customize it to your specific simulation. This file contains the subroutines `main()`, `sdumotion()`, `sduforce()` and `init_motion()`.

General Purpose Source Files

`/software/simm2/pipeline/src/gmc.c`

General muscle code – a collection of routines to calculate musculo-tendon model quantities (e.g. passive muscle force, tendon force and pennation angle) from current state information

`/software/simm2/pipeline/src/mathtools.c`
General purpose mathematics routines

`/software/simm2/pipeline/src/pipetools.c`
Utility routines for conducting dynamic simulations.

`/software/simm2/pipeline/src/readmuscles.c`
Routines to read in the musculo-tendon model information from a muscle file. Attachment points, wrapping surfaces, and muscle-specific parameters (e.g. maximum isometric force, tendon slack length, etc…) are input and use during the simulation. Muscle files are read at runtime allowing the changing of muscle parameters and muscle excitations without altering the simulation code.

`/software/simm2/pipeline/src/readtools.c`
General purpose utility routines used to read in data from input muscle files and kinetics data files.

Muscle-Tendon Model Source Files

`/software/nmbl/tutorials/me382/L4/assigns.c`
`/software/nmbl/tutorials/me382/L4/derivs.c`
`/software/nmbl/tutorials/me382/L4/inits.c`
Code implementing dynamic muscle-tendon models. The code is set up such that you can use one of the existing models described in the Pipeline manual or can alternatively use templates that are provided to create your own model.

Header Files

`/software/simm2/pipeline/src/basic.h`
Contains #defines and enum that are used in many source files.

`/software/simm2/pipeline/src/functions.h`
Contains prototypes for many of the functions in the source files.

`/software/simm2/pipeline/src/structs.h`
Contains definitions of all of the structures that are used in the Dynamics Pipeline.

`/software/simm2/pipeline/src/universal.h`
Contains #includes for all of the standard header files. This file should be included at the top of every source file in the Pipeline.

Make Files

`/software/simm2/pipeline/src/Makefile.forward`
The makefile for a forward dynamics simulation. It assumes that the name of your SD/FAST system description file is model.sd and the name of your model specific C file is sdfor.c. Can be run by entering `#make -f Makefile.forward`.

Libraries

3

```
/software/simm2/pipeline/n32libs/libacpp.a
```
General purpose routines used for parsing files on input.

```
/software/simm2/pipeline/n32libs/libwrap.a
```
Precompiled library that accounts for muscle wrapping in the calculation of muscle-tendon length and velocity during a dynamic simulation.

V.   Getting Started

Copy all of the input files into a directory that you create for this lab.
```
# mkdir L4
# cd L4
# cp /software/simm2/pipeline/src/* .
# cp /software/nmbl/tutorials/me382/L4/* .
# cp /software/simm2/pipeline/n32libs/* .
# mkdir bones
# cd bones
# cp /software/nmbl/tutorials/me382/L4/bones/* .
```

Note: There are different versions of the muscle-tendon model routines (`assigns.c`, `derivs.c`, `inits.c`) in two of the directories accessed above (`/software/nmbl/tutorials/me382/L4`, `/software/simm2/pipeline/src`). In this lab, you will be using the routines in `/software/nmbl/tutorials/me382/L4`, which contains a slightly modified version of model 4 described in the Pipeline tutorial.

VI.   Muscle and Tendon Modeling

Thorough review articles have been written on the development and use of Hill-type musculo-tendon models in dynamic simulations of movement [Zajac 1989, Winters 1990]. Following is a brief review of the Hill-type model put forth in those papers, as it has been implemented within SIMM pipeline. If you are interested in greater detail, you should consult the references directly.

VI-A. *Activation dynamics*

A muscle is not capable of generating force or relaxing instantaneously. The development of force is a complex sequence of events, which begins with the firing of motor units and culminates in the formation of actin-myosin cross-bridges within the myofibrils of the muscle. When the motor units of a muscle depolarize, action potentials are elicited in the fibers of the muscle and cause calcium ions to be released from the sarcoplasmic reticulum. The increase in calcium ion concentrations then initiates the cross-bridge formation between the actin and myosin filaments (See Guyton (1986) for review). In isolated muscle twitch experiments, the delay between a motor unit action potential and the development of peak force has been observed to vary from as little as 5 milliseconds for fast ocular muscles to as much as 40 or 50 milliseconds for muscles comprised of higher percentages of slow-twitch fibers. The relaxation of muscle depends on the re-uptake of calcium ions into the sarcoplasmic reticulum. This re-uptake is a slower process than the calcium ion release, and so the time required for muscle force to fall can be considerably longer than the time for it to develop.

In the muscle simulations you will conduct in this lab, activation dynamics is modeled using a first-order differential equation with a variable time constant. This equation relates the rate of change in activation (i.e., the concentration of calcium ions within the muscle) to excitation (i.e., the firing of motor units):

$$\dot{a} = \frac{(x - a)}{\tau(a,x)} \qquad\qquad (1)$$

where a is the activation level of a muscle, x is the excitation level and τ is a variable time constant which is given by:

$$\tau(a,x) = \begin{cases} (\tau_{act} - \tau_{deact})x + \tau_{deact} & x > a \\ \tau_{deact} & x < a \end{cases} \qquad\qquad (2)$$

In the above equation, the parameters describe the rates of rise of activation (τ_{act}) and deactivation (τ_{deact}) in response to muscle excitation. In the model, activation is allowed to vary continuously between zero (no contraction) and one (full contraction). In the body, the excitation level of a muscle is a function both of the number of motor units recruited and the firing frequency of the motor units. Some models for excitation-contraction coupling distinguish these two control mechanisms (Hatze, 1976), but it is often not computationally feasible to use such models when conducting complex dynamic simulations. In the simulation, the muscle excitation signal is assumed to represent the net effect of both motor neuron recruitment and firing frequency, and, like muscle activation, is also allowed to vary continuously between zero (no excitation) and one (full excitation). The activation and deactivation time constants can be assumed to be 15 and 50 ms, respectively (Zajac, 1989, Winters 1990).

VI-B Muscle-tendon contraction dynamics

The force producing properties of muscle are complex and nonlinear (See McMahon 1984 for review). For simplicity, lumped-parameter dimensionless muscle models, capable of representing a range of muscles with different architectures, are commonly used in dynamic simulation of movement (Zajac, 1989). In this model, the muscle force-length and force-velocity, and tendon force-strain relationships are represented by dimensionless curves (Figure 1).
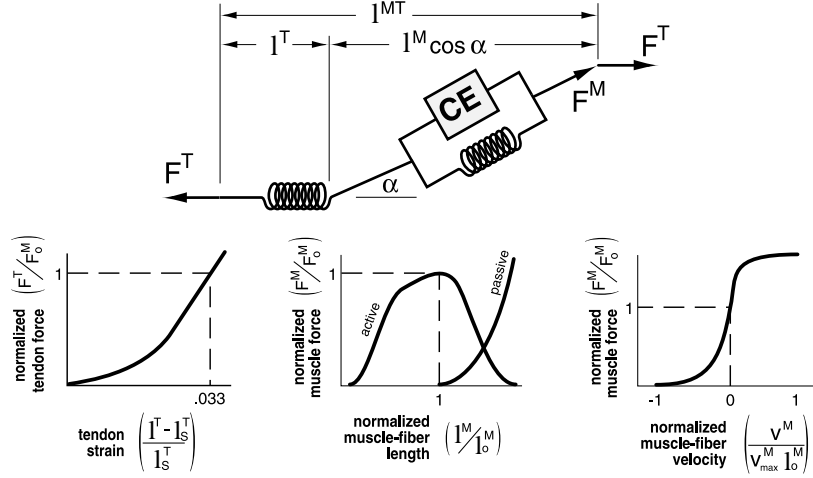
Figure 1. Dimensionless model of muscle and tendon. Muscle properties are represented by an active contractile element (CE) in parallel with a passive elastic element *(top)*. Muscle force is dependent on muscle fiber length *(middle plot)* and velocity *(right plot)*. Muscle is in series with tendon, which is represented by a nonlinear elastic element *(left plot)*. Pennation angle (α) is the angle between the muscle fibers and the tendon. The forces in muscle and tendon are normalized by peak isometric muscle force (F_o^M). Muscle fiber length (l^M) and tendon length (l^T) are normalized by optimal fiber length (l_o^M). Tendon slack length (l_S^T) is the length at which tendons begin to transmit force when stretched. Velocities are normalized by the maximum contraction velocity of muscle (V_{max}^M). For a given muscle-tendon length (l^{MT}), velocity, and activation level, the model computes muscle force (F^M) and tendon force (F^T).

Four muscle-specific parameters are commonly used to scale the dimensionless curves for individual muscles:

- F_0^M maximum isometric muscle force
- l_0^M optimal muscle fiber length
- l_S^T tendon slack length and
- α pennation angle

There are different choices of state that can be used for the muscle-tendon model. Either muscle-tendon force (Zajac 1989, Anderson and Pandy 1999) or tendon length (Winters 1990) is commonly used. In this lab, you will be using muscle length, which has been shown to have the advantage of not requiring inversion of the tendon force-strain curve (Schutte 1992). Thus the state equation for musculo-tendon dynamics can be given by:

$$\dot{l}^M = v^M(\vec{q}, \dot{\vec{q}}, a, l^M) \qquad\qquad (3)$$

where $\vec{q}$ represents the generalized coordinates of the system, $\dot{\vec{q}}$ the vector of time derivatives of the generalized coordinates, a is the muscle activation and l^M is the current muscle fiber length. During a forward dynamic simulation, muscle length and activation are treated as states and solved for by numerically integrating equations (1) and (3) simultaneously with the system equations of motion. Within a simulation, the muscle-tendon force (F^{MT}) acting on the skeleton is computed from the system states: muscle length, activation, generalized coordinates and generalized speeds. The generalized coordinates and generalized speeds of the system are first used to numerically compute the muscle-tendon length (l^{MT}) and muscle-tendon velocity ($\dot{l}^{MT}$). For example, muscle-tendon length is computed by adding up the incremental lengths between muscle path points as defined in a joint file. Checks are included to see if wrapping surfaces are being contacted by the muscle and if so, the additional length required to wrap around a surface is included in the computation of the muscle-tendon length. Muscle-tendon velocities are computed similarly by adding up the incremental velocities between muscle path points as defined in a joint file. After computing l^{MT} and $\dot{l}^{MT}$, the muscle activation is used along with the dimensionless curves shown in Figure 1, to compute the muscle-tendon force.

VI-C   SIMM/Pipeline Implementation of Muscle-Tendon Model

Various versions of the Hill-type muscle-tendon models, as discussed previously, have been implemented in the SIMM Pipeline code. There are a few notes that should be made with regards to the implementation. First, a normalized form of the contraction dynamics state equation is used. All length quantities are normalized to optimal muscle fiber length l_0^M, force quantities are normalized to maximum isometric force F_0^M and time is normalized to the inverse normalized maximum contraction velocity $\tau_c = 1/v_{max}^M$. After normalization, the contraction dynamics state equation is written as

$$\dot{\tilde{l}}^M = \tilde{v}^M(\vec{q}, \dot{\vec{q}}, a, \tilde{l}^M)$$

where the normalized muscle length is given by $\tilde{l}^M = l^M / l_0^M$.

Solving the state equation requires inversion of the force-velocity curve, which can be problematic when either muscle activation is near zero or the muscle fibers are very long or very short which results in a small active force-length factor. This difficulty is overcome in the model by including a small amount of passive damping in parallel with the contractile element such that the force-velocity curve remains invertible at low activation (Schutte 1992).

The Pipeline code (`assigns.c, derivs.c, inits.c`) can include up to 10 different muscle-tendon models, numbered 1 through 10. In this lab, you will be using muscle model #7, which is muscle model #4, which is described in the Pipeline manual, with a slight change in model parameters to make them more intuitive. The dynamic muscle parameters that must be defined in the muscle file are:

- *timescale* –inverse of the normalized maximum contraction velocity [$\tau_c = 1/\tilde{v}_{max}^M$]. $\tilde{v}_{max}^M$ is expressed in fiber lengths per second $\tilde{v}_{max}^M = v_{max}^M / L_0^M$.

- *activation_timeconstant* – muscle activation time constant $\tilde{\tau}^{act}$]

- *deactivation_timeconstant* – muscle deactivation time constant [τ_{deact}]

- *damping* – normalized passive damping in parallel with contractile element [$\overline{b} = \dfrac{b}{F_0^M / L_0^M \, \overline{v}_{max}}$]

VII. Using SIMM/Pipeline to Simulate Muscle-Tendon Dynamics

VII-A SIMM/Pipeline Basics

The basic steps involved in creating a muscle driven simulation using SIMM Pipeline and SD/FAST involve:

1. Using SIMM/Pipeline to generate an SD/FAST system description and model specific code (sdfor.c) from a joint file.

2. Running SD/FAST to process the system description file and generate code that represents the equations of motion of the system.

3. Incorporate SIMM/Pipeline and SD/FAST routines with any additional custom code you desire to simulate your particular application. Often custom code is used to simulate external contact forces, to run a sensitivity study or to perform numerical optimization. A driver program, formain.c, is generated by SIMM/Pipeline and can be used as a starting point for creating an application specific driver routine for your simulation.

As a preview, please read the following sections of the Dynamics Pipeline manual.

Chapter 1   Introduction

Chapter 2   Input Files

Chapter 3   Muscle Modeling

Chapter 4   Forward Dynamics Analysis

Note that Chapter 2 describes the additional information that must be included in a SIMM joint and muscle file in order to create a dynamic simulation. In a joint file, this additional information includes the mass and moments of inertia of body segments in your model. In a muscle file, dynamic muscle-tendon model parameters are added that are used in the activation and dynamic contraction equations. In addition, the muscle file can include a description of the muscle inputs or excitations. Either open-loop step patterns or spline fits can be used to specify excitation as a function of time or as a function of a generalized coordinate.

VII-B Muscle Tug of War Model

In this lab, you will be using a simple mechanical model to investigate the dynamics properties of muscle-tendon actuators. The model consists of a block translating on a frictionless surface under the action of two opposing muscle-tendon actuators, i.e. a *muscle tug of war* (Figure 2).
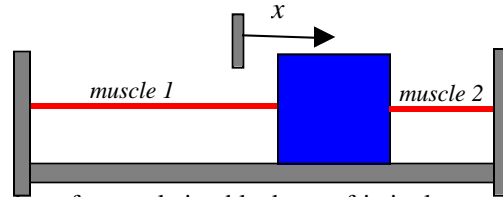
Figure 2. Model consists of a translating block on a frictionless surface being acted upon by two opposing muscles. The block is 0.1 x 0.1 m and has a mass of 20 kg. The distance between fixed supports is 0.7 meters. With the block centered, the muscle-tendon lengths are 0.3 m.

The following steps should be followed to create a dynamic, muscle-driven simulation of the translating block model.

1. Start SIMM and load the joint and muscle files
    Change to lab 4 directory :   `# cd L4`
    Start SIMM: `# simm2`
    Load the muscle and joint files:
    Joint file: `tug_of_war.jnt`
    Muscle file: `tug_of_war.msl`

2. Use SIMM Pipeline to write dynamics files for your system. Open the `File Writer` tool and click on the `forward dyn` button. SIMM Pipeline will create two files in your current working directory (the directory where you started SIMM).
    *   `model.sd`   SD/FAST system description file that is used to generate code that describes the system equations of motion
    *   `sdfor.c`   Model specific C code that contains the body segment parameters and joint kinematics

3. Set the default parameter values that appear in `forparams.txt`. `Forparams.txt` contains the names of the input and output files that the simulation needs. The following file names should be set:
    *   `muscle_file`   `tug_of_war.msl`
    *   `output_motion_file`   `forward.mot`

4. Compile the simulation program using the `makefile` provided. The `makefile` includes a SD/FAST call that processes your system description file. Therefore as in the last tutorial, you need to be logged on to `Hill` for the makefile to be able to run SD/FAST.

    `# make -f Makefile.forward`

    Don't be surprised to see a few warnings the first time you compile your code. The SD/FAST library (`sdlib.c`) does not have an accompanying header file and consequently you will get warnings about functions not being previously declared. Running the `makefile` will create an executable file called `sdfor`. By default, the `makefile` compiles the code using the debugging option (`-g`). Once you have confirmed that the simulation is running properly, you can change this option to `-O2` to make the program run faster.

9

5. Run the simultion

   ```
   # sdfor
   ```

6. View the simulation results in SIMM. Load the motion file `forward.mot` and animate the motion. You may need to set the gear (speed) of the animation slower for the animation to proceed at a speed that you can visualize. The sizes and colors of the muscles will change to reflect their activation levels. Use the `motion curve >` command in the `Plot Maker` to make graphs of the generalized coordinate x and the muscle activations. You should get results that look like those shown in Figure 3a and b.

7. Chapter 2 of the Pipeline manual describes how the muscle excitation-time relationship can be specified as either an open-loop step function or spline fit. Sketch out the excitations of muscles A and B that were used to generate the previous simulation.

8. A useful feature of defining the muscle parameters and excitations within the muscle file is the ability to make changes to the model parameters or inputs without recompiling the program. You will be using these capabilities extensively in designing your muscle for the tug of war. Practice changing the excitation functions and rerun the simulations to demonstrate how the simulations respond to difference excitation patterns.
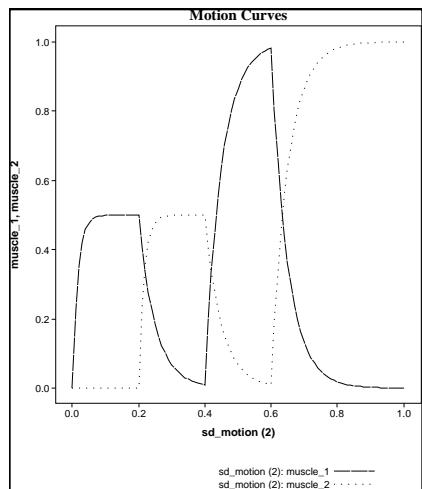
(a)   (b)   (c)

Figure 3. Muscle activations (a), block translation (b) and muscle forces (c) for initial demonstration simulation.

VIII. Exploration Phase

The objective of the following sets of exercises is for you to gain experience using and modifying SIMM Pipeline simulation code. You will also begin to look at how various muscle model parameters affect the dynamic response of a muscle-tendon actuator, insight that should aid your design of your *optimal* muscle for the virtual tug of war.

VIII-A Exploration of SIMM Pipeline Code

1. As mentioned previously, muscle length is a state variable that is solved for by numerical integration of the contraction dynamics equation. However tendon force is what must be applied to the mechanical system during an actual simulation. Investigate the pipeline code and document how the transformation from muscle length to muscle-tendon force is made. Write down all the routines that are called and explain how the curves shown in Figure 1 are actually implemented in the code to make the transformations from muscle length to tendon force.

2. By further exploring the code, explain how muscle-tendon forces are actually applied to the skeleton within a dynamic simulation.

3. Often we are interested in knowing the net muscle moment about a joint, since that quantity can be compared with joint moments computed using inverse dynamics. How would you go about computing net muscle moments about a joint using SD/FAST and/or Pipeline code?

VIII-B Modification of SIMM Pipeline Code

Currently `formain.c` only outputs muscle activation and the time derivatives of muscle activation to the motion file. However you might well be interested in plotting muscle-tendon force. Modify the driver routine `formain.c` to output headers for and the values of muscle-tendon forces. In the header, name the muscle forces `muscle_1_force` and `muscle_2_force`. To access muscle forces, you may want to look in the file `structs.h` to see how muscle force is saved in a muscle structure. Using the simulation of the previous section, you should get muscle-tendon forces that look like those shown in Figure 3c.

VIII-C Affect of Model Parameters on Dynamic Response

Zajac (1989) showed that the tendon-to-fiber length ratio (l_S^T / l_0^M) can have substantial effects on the mechanical response of a muscle-tendon actuator. The following sets of simulations are designed to have you vary the simulation code and parameters in order to examine the effect of tendon-to-fiber length ratio on the mechanical response of a muscle during isometric and isokinetic contractions. For each of these simulations, comment out muscle_1 in the muscle file such that the simulation includes only one muscle (muscle_2).

1. <u>Isometric Responses:</u> Perform a set of simulations in which you vary the tendon-to-fiber length ratio and look at how this affects the force-time response in an isometric simulation. For these simulations, fully activate muscle_2 at time t=0. Modify the `formain.c` program such that the block does not move (use prescribed motion in SD/FAST) and muscle_2 contracts isometrically. Perform isometric simulations with a range of tendon-to-fiber length ratios: l_S^T / l_0^M = 0.5, 1.0, 2.0, 4.0, 8.0. For each tendon-to-fiber length ratio, maintain the sum of the slack tendon length and optimal fiber length constant ($l_S^T + l_0^M = 0.3m$).

Overly curves of the muscle force-time histories for each tendon-to-fiber length ratio. Describe any differences that you see between the curves in terms of the rate of force development and steady state force achieved. Using what you know about muscle activation dynamics and muscle-tendon mechanics, explain why the muscle force responses differ the way they do with changes in tendon-to-fiber length ration.

2. Isokinetic Responses: Isokinetic exercises (constant velocity) are often used to characterize the force generating properties of muscle. For example, isokinetic dynamometers are used to measure the maximum joint moment a human can produce during constant a constant angular velocity contraction. In this set of simulations, you will analyze how the output of a muscle-tendon actuator varies with tendon-to-fiber length ratio during an isokinetic contraction. You should modify your formain.c driver program such the block translates at a constant *0.1 m/s* to the right after starting an initial position *0.05 m* to the left of the mid-position. Use prescribed motion in SD/FAST to control the motion of the block. The initial block position can be varied in `sdfor.c`. Comment out `muscle_1` from the muscle file and fully activate `muscle_2` at time `t=0`. Perform isometric simulations with the following tendon-to-fiber length ratios: l_S^T / l_0^M = *0.5, 1.0, 2.0.* For each tendon-to-fiber length ratio, maintain the sum of the slack tendon length and optimal fiber length constant ($l_S^T + l_0^M = 0.3m$).

Overly plots of the muscle force as a function of time from the three simulations. Describe any differences that you see between the curves in terms of the magnitude and timing of peak force, and the shape of the curve. Explain why the simulations vary the way they do. Your explanation should describe how the muscle force-length-velocity and tendon force-strain properties combine to give the response you see.

IX.   Design Phase

Virtual Muscle Tug of War: A single elimination muscle tug-of-war tournament will be held between members of the class. A match will consist of two muscles competing head-to-head (muscle-to-muscle) in a one second, winner-take-all tug of war. You will be required to specify the muscle-tendon parameters and excitation-time history subject to the constraints described below. In each tug of war, whichever muscle has moved the block onto their side at the end of one second is declared the winner. Both muscles will start a match at rest with zero activation.

Design variables – you must specify the values of the following variables

F_0^M maximum isometric muscle force

τ_C inverse of the normalized maximum contraction velocity$[=1/v_{max}^M]$

l_0^M optimal muscle fiber length

α_0 muscle fibre pennation angle corresponding to muscle fiber at optimal length

$x(t)$ muscle excitation time history

Definition of other model parameters you will use

A^M physiological cross-sectional area of muscle in cm^2

V^M muscle volume ($=A^M l_0^M$)

σ_0^M maximum isometric stress of muscle, assumed equal to *35 N/cm^2* (Zajac 1989)

Design constraints

$F_0^M = A^M \sigma_0^M$

$V^M \leq 100\ cm^3$

$0.05\,m \leq l_0^M \leq 0.2\ m$

$l_S^T \geq 0.1\ m$

$0.0 \leq \alpha_0 \leq 30°$

$2 \leq v_{max}^M \leq 10$

$F_0^M\, v_{max}^M\, l_0^M \leq 175\ W$

$\int_0^1 x(t)dt \leq 0.5$

$10\ ms \leq \tau_{act} \leq 20\ ms$

$40\ ms \leq \tau_{deact} \leq 60\ ms$

$30\ ms \leq \tau_{deact} - \tau_{act} \leq 40\ ms$

Additional Note: Passive muscle forces will not be included in the tug of war. In performing simulations as part of your muscle design process, you are responsible for turning off muscle passive force. To do this, edit the file `derivs.c`. Under `musc_deriv_func7`, replace the following line:

```
passive_force = calc_nonzero_passive_force(ms,normstate[fiber_length],0.0);
```

with:

```
passive_force = 0.0;
```

Design Process:

You are individually responsible for devising and implementing a process to use in designing the muscle-tendon actuator. For this class, the process and results should be clearly documented in a brief report. The report, excluding figures and tables, should not be more than 2 pages. It should include the following sections:

- Introduction
  - o Objectives of your design
- Methods
  - o Outline of steps used in your design process
- Results
  - o Should include a combination of mathematical analysis, parameter sensitivity studies and results of prototype muscle simulations.
  - o Description of the final muscle design
- Discussion
  - o Justification of your final design
  - o Evaluation of the strengths and potential weaknesses of your design, e.g. under what conditions do you expect your muscle to perform well?

X. References

Anderson FC and Pandy MG (1999). A dynamic optimization solution for jumping in three dimensions. *Computer Methods in Biomechanics and Biomedical Engineering*, **2**, 201-231.

Delp SL, Loan P (1995) A graphics-based software system to develop and analyze models of musculoskeletal structures. *Comput Biol Med* 1:21-34.

Hatze H (1976). The complete optimization of human motion. *Mathematical Biosciences*, **28**, 99-135.

McMahon TA (1984). Muscles, Reflexes, and Locomotion. Princeton University Press, Princeton, New Jersey.

Schutte LM (1992), Using Musculoskeletal Models to Explore Strategies for Improving Performance in Electrical Stimulation-Induced Leg Cycle Ergometry. Ph.D. Dissertation, Mechanical Engineering Department, Stanford University.

Symbolic Dynamics, Inc. (1996). SD/FAST User's Manual, Version B.2. Mountain View, CA.

Winters JM, 1990, "Hill-based muscle models: a systems engineering perspective," in Multiple Muscle Systems: Biomechanics and Movement Organization, edited by JM Winters and SL Woo, Springer-Verlag, New York.

Zajac FE (1989). Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. *CRC Critical Reviews in Biomedical Engineering*, **17**, 359-411.